

FACIAL EMOTION DETECTION AND RECOGNITION

A MINI PROJECT BY

SUBMITTED BY:

GOWTHAM.R [511320104023]

HUBERT RYAN.E [511320104026]

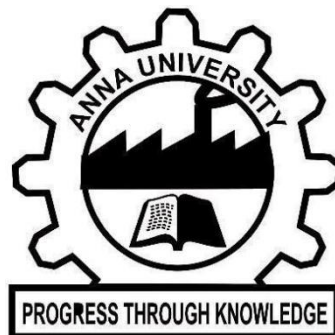
DHANUSH.N [511320104017]

In partial fulfillment for the award of the Degree of

BACHELOR OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING

KINGSTON ENGINEERING OF COLLEGE, VELLORE

ANNA UNIVERSITY: CHENNAI 600 025



JUNE 2023

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report “**Facial Emotion Detection and Recognition**” is the bonafide of “**R.Gowtham–511320104023,E.Hubert Ryan – 511320104026 , N.Dhanush – 511320104017 ”** who carried out the project work under my supervision.

SIGNATURE

Dr.U.V.ARIVAZHAGU M.E.,Ph.D,
HEAD OF THE DEPARTMENT

Computer Science Engineering,
Kingston Engineering College,
Vellore – 632 059

SIGNATURE

Ms.C.CHINNIMA M.E
SUPERVISIOR

Assistant Professor,
Computer Science Engineering,
Kingston Engineering College,
Vellore– 632 059

Submitted for the project viva voice examination held on

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

First and foremost, we extremely grateful to the **almighty** giving the strength and support to complete this project successfully.

We feel happy to express us profound gratitude to our honorable chairman **Thiru.D.M.KathirAnand M.B.A., (U.S.A)** who provided all the facilities to us and our principal as well as Head of the Department **Dr.U.V.Arivazhagu M.E.,Ph.D.** who gave us a chance to quench our knowledge thirst and also who have been kind to us in all aspects.

We express thanks to our project coordinator **Mr.N.Karthik M.Tech**, Assistant Professor, for his advice and guidance to complete my project work. We thank all our **faculty members** who have helped us in the completion of this project. With immense pleasure we regard our deep sense and our gratitude to our project guide **Ms.C.Chinnima M.E**, Assistant Professor, Department of Computer Science and Engineering, who was not only a source of inspiration but also motivating us with her guidance and continuous support in the execution of the project.

To one and all, we own acknowledgement who directly or indirectly aided us in completing the project.

ABSTRACT

Facial emotional expression is a part of face recognition, it has always been an easy task for humans, but achieving the same with a computer algorithm is challenging. With the recent and continuous advancements in computer vision and machine learning, it is possible to detect emotions in images, videos, etc. A face expression recognition method based on the Deep Neural Networks especially the convolutional neural network (CNN) and an image edge detection is proposed. The edge of each layer of the image is retrieved in the convolution process after the facial expression image is normalized. To maintain the texture picture's edge structure information, the retrieved edge information is placed on each feature image. In this research, several datasets are investigated and explored for training expression recognition models. The purpose of this paper is to make a study on face emotion detection and recognition via Machine learning algorithms and deep learning. This research work will present deeper insights into Face emotion detection and Recognition. It will also highlight the variables that have an impact on its efficacy.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	IV
	LIST OF ABBREVIATIONS	IX
	LIST OF FIGURES	X
1.	INTRODUCTION	1
	1.1 Objective	1
	1.2 Overview	1
	1.3 Introduction	1
2	LITERATURE SURVEY	3
3	SYSTEM ANALYSIS	10
	3.1 Existing System	10
	3.1.1 Disadvantages	10
	3.2 Proposed System	11
	3.2.1 Advantages	11
4	SYSTEM REQUIREMENTS	12
	4.1 Hardware Requirements	12
	4.2 Software Requirements	12
	4.3 Technologies Uesd	13
	4.3.1 Java Technology	13
	4.3.1.1 Java Programming	13
	4.3.1.2 Java Platform	14
	4.3.1.3 ODBC	18
	4.3.1.4 JDBC	20
	4.3.1.5 Networking	23
	4.3.1.6 JFree Chart	26
	4.3.1.7 J2ME	28

5	SYSTEM DESIGN SPECIFICATION	32
	5.1 Architecture Diagram	32
	5.2 Data Flow Diagram	32
	5.3 Use Case Diagram	33
	5.4 Activity Diagram	34
	5.5 Sequence Diagram	37
6	SYSTEM DESIGN IMPLEMENTATION	38
	6.1 Implementation	38
	6.2 Modules Description	38
	6.2.1 Database Description	38
	6.2.1.1 Pre-Processing	40
	6.2.1.2 Training Module	40
	6.2.2 Emotion Detection	40
	6.2.2.1 Face Detection	40
	6.2.2.2 Feature Extration	41
	6.2.2.3 Emotion Detection	42
7	SYSTEM TESTING AND MAINTENANCE	45
	7.1 Testing Objective	45
	7.2 Testing And Methodology	46
	7.2.1 Unit Testing	46
	7.2.2 Functional Testing	46
	7.2.3 System Testing	47
	7.2.4 White BoxTesting	47
	7.2.5 Black Box Testing	47
	7.2.6 Integration Testing	48
	7.2.7 Acceptance Testing	48

8	CONCLUSION & FUTURE ENHANCEMENT	49
	8.1 Conclusion	49
	8.2 Future Enhancement	49
	APPENDIX I	50
	Source Code	
	APPENDIX II	57
	Output Processors	
	REFERENCES	62

LIST OF ABBREVIATIONS

FER	FACIAL EMOTION RECOGNITION
CNN	CONVOLUTIONAL NEURAL NETWORK
DNN	DEEP NEURAL NETWORK
HCCA	HAAR CASCADE ALGORITHM

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
5.1	Architecture Diagram	32
5.2	Data Flow Diagram	32
5.3	Use Case Diagram	33
5.4	Activity Diagram	34
5.5	Sequence Diagram	37

CHAPTER 1

INTRODUCTION

1.1 OBJECTIVE

The main objective is to design a suitable facial emotion recognition system to identify various human emotional states by analyzing facial expressions which is used to improve Human-Computer Interaction (HCI). This is achieved by analyzing hundreds of images of different emotional states of a human being. The analysis is carried out by a computer system which uses different technologies to recognize and study the different emotional states.

1.2 OVERVIEW

Facial Emotion Recognition (FER) is the technology that analyses facial expressions from both static images and videos in order to reveal the information on ones emotional states.

1.3 INTRODUCTION

Human-computer interaction technology refers to a kind of technology that takes computer equipment as the medium, so as to realize the interaction between humans and computers. Face recognition system (FRS) is a mechanism that allows cameras to automatically identify people. Because of the importance of correct and effective FRS, it drives the activeness of biometric research in the race to the digital world.[8] In recent years, with the rapid development of pattern recognition and artificial intelligence, more and more research has been conducted in the field of humancomputer interaction technology. [1] Facial Emotion Recognition (FER) is a flourishing study topic in which many breakthroughs are being made in industries, such as automatic translation systems and machine-to-human contact. In contrast, the paper focus to survey and reviewing various facial extraction features, emotional databases, classifier algorithms, and so on.[4] The classical FER consists of two main steps: feature extraction and emotion recognition. In addition, image pre-processing,

including face detection, cropping, and resizing. Face detection crops the facial region after removing the backdrop and non-face areas. Finally, the retrieved characteristics are used to classify emotions, which is commonly done with the help of neural networks (NN) and other machine learning approaches. The challenge of facial emotion recognition is to automatically recognize facial emotion states with high accuracy. Therefore, it is challenging to find the similarity of the same emotional state between different people since they may express the same emotional state in various ways. As an example, the expression may vary in different situations such as the individual's mood, skin colour, age, and the environment surrounding. Generally, FER is separated into three major stages as shown in Figure 1: (i) Face Detection, (ii) Feature Extraction, and (iii) Emotion Classification. Face detection refers to the computer vision technique of locating and identifying human faces within images or video streams. It is an essential component in many applications, including facial recognition, emotion analysis, biometrics, augmented reality, and more. The primary goal of face detection is to determine the presence and location of faces within an input image or video frame. It involves analyzing the visual patterns, shapes, and features that define a face, and then identifying regions that are likely to contain faces. Face detection algorithms typically operate by scanning an image or video frame and searching for patterns that resemble facial features. These algorithms utilize various techniques, such as machine learning, image processing, and computer vision, to identify facial characteristics such as eyes, nose, mouth, and other facial landmarks. Modern face detection methods often employ deep learning models, such as convolutional neural networks (CNNs), to achieve high accuracy and robustness. These models are trained on vast amounts of annotated data to learn and recognize patterns indicative of faces.

CHAPTER2

LITERATURE SURVEY

[1] : TITLE: “FACIAL EMOTION RECOGNITION: A BRIEF REVIEW”

AUTHORS: Illiana Azizan, K. Fatimah

JOURNAL NAME AND YEAR: IEEE CONFERENCE,2020

DESCRIPTION:

This paper give an overview of current Facial Emotion Recognition (FER) stages, techniques, and datasets. FER has been recognized for a decades and it is a vital topic in the fields of computer vision and machine learning. Automatic FER is useful in most of the applications such as healthcare, teaching, criminal investigation, Human Robot Interface (HRI), etc. This paper is aim to understand the basic principles of FER and make a comparison of current research.

LIMITATIONS:

- It only works in initial and target problems.
- The relative angle of the target’s face has a significant impact on the recognition
- Emotions can be subjective and vary across individuals and cultures.
- Facial expressions are not solely indicative of emotions but can also be influenced by various contextual factors.

**[2]:TITLE : “MACHINE LEARNING AND ITS
DOMINANT PARADIGMS”/**

AUTHORS: Radhey Shyam, Riya Chakraborty

JOURNAL NAME AND YEAR: 2021

DESCRIPTION:

Much of the human behavior depends on learning. This learning begins at a very small age in humans when a newborn start understanding what is happening around. They get to describe what is happening which results into descriptive learning. When they analyze the cause or nature of a condition, it is called diagnostic learning. When they validate the foreseen probability of things, that is predictive learning. They tend to create definitive activities from their learning which is perspective learning. Finally, as a grown adult, when after all the learning, their actions become intuitive, that is cognitive learning. Just as humans, machines also have the capacity to learn through experience and training. Machines also have the capabilities to learn and predict, form perceptions, and take actions accordingly. All these falls under the domain of Machine Learning. Therefore, the paper will present deeper understanding of machine learning and its paradigms as well.

LIMITATIONS:

- CNN tends to be much slower because of operations like MAXPOOLING.
- The effectiveness of facial-recognition algorithms is influenced by the image quality.

**[3] :TITLE:"FACIAL EMOTION RECOGNITION USING TRANSFER
LEARNING IN THE DEEP CNN"**

AUTHORS: M. A. H. Akhand 1,*, Shuvendu Roy 1

JOURNAL NAME AND YEAR: 2021

DESCRIPTION:

Human facial emotion recognition (FER) has attracted the attention of the research community for its promising applications. Mapping different facial expressions to the respective emotional states are the main task in FER. The classical FER consists of two major steps: feature extraction and emotion recognition. Currently, the Deep Neural Networks, especially the Convolutional Neural Network (CNN), is widely used in FER by virtue of its inherent feature extraction mechanism from images. Several works have been reported on CNN with only a few layers to resolve FER problems. However, standard shallow CNNs with straightforward learning schemes have limited feature extraction capability to capture emotion information from high-resolution images. A notable drawback of the most existing methods is that they consider only the frontal images (i.e., ignore profile views for convenience), although the profile views taken from different angles are important for a practical FER system. For developing a highly accurate FER system, this study proposes a very Deep CNN (DCNN) modeling through Transfer Learning (TL) technique where a pre-trained DCNN model is adopted by replacing its dense upper layer(s) compatible with FER, and the model is fine-tuned with facial emotion data. A novel pipeline strategy is introduced, where the training of the dense layer(s) is followed by tuning each of the pre-trained DCNN blocks successively that has led to gradual improvement of the accuracy of FER to a higher level. The proposed FER system is verified on eight different pre-trained DCNN models (VGG-16, VGG-19, ResNet-18, ResNet-34, ResNet-50, ResNet-152, Inception-v3 and DenseNet-161) and well-known KDEF and JAFFE facial image datasets. FER is very challenging even for frontal views alone. FER on the KDEF dataset poses further challenges due to the diversity of images with different profile views together with frontal views. The

proposed method achieved remarkable accuracy on both datasets with pre-trained models. On a 10-fold cross-validation way, the best achieved FER accuracies with DenseNet-161 on test sets of KDEF and JAFFE are 96.51% and 99.52%, respectively. The evaluation results reveal the superiority of the proposed FER system over the existing ones regarding emotion detection accuracy. Moreover, the achieved performance on the KDEF dataset with profile views is promising as it clearly demonstrates the required proficiency for real-life applications.

LIMITATIONS:

- Poor image quality, small image size and data processing and storage.
- It has difficult operating systems.
- Manual tuning of parameters.
- Transfer learning relies on pre-trained models that have been trained on large-scale datasets for general object recognition tasks.
- Pre-trained models used in transfer learning are typically trained on diverse datasets. However, facial emotion datasets may have different distributions and biases, such as variations in age, ethnicity, gender, or cultural backgrounds.
- Fine-tuning is an important step in transfer learning, where the pre-trained model is adapted to the target task using a smaller dataset.
- Facial emotions are complex and can involve a combination of facial expressions, subtle cues, and contextual factors.
- Transfer learning with deep CNNs may struggle with generalizing to unseen or novel emotions that are not well-represented in the training data.
- Deep CNNs, even with transfer learning, can be computationally demanding, requiring significant processing power and memory.
- Facial emotion recognition raises ethical concerns related to privacy and consent. Collecting and analyzing individuals' facial expressions without their informed consent or for unethical purposes can infringe on privacy rights and raise ethical considerations.

[4] TITLE:”FACIAL EMOTION RECOGNITION USING CONVOLUTIONAL NEURAL NETWORKS (FERC)”

AUTHORS:Ninad Mehendale

JOURNAL NAME AND YEAR:2020

DESCRIPTION:

Facial expression for emotion detection has always been an easy task for humans, but achieving the same task with a computer algorithm is quite challenging. With the recent advancement in computer vision and machine learning, it is possible to detect emotions from images. In this paper, we propose a novel technique called facial emotion recognition using convolutional neural networks (FERC). The FERC is based on two-part convolutional neural network (CNN): The first-part removes the background from the picture, and the second part concentrates on the facial feature vector extraction. In FERC model, expressional vector (EV) is used to find the five different types of regular facial expression. Supervisory data were obtained from the the emotion with 96% accuracy, using a EV of length 24 values. The two-level CNN stored database of 10,000 images (154 persons). It was possible to correctly highlight works in series, and the last layer of perceptron adjusts the weights and exponent values with each iteration. FERC differs from generally followed strategies with single-level CNN, hence improving the accuracy. Furthermore, a novel background removal procedure applied, before the generation of EV, avoids dealing with multiple problems that may occur (for example distance from the camera). FERC was extensively tested with more than 750K images using extended Cohn–Kanade expression, Caltech faces, CMU and NIST datasets. We expect the FERC emotion detection to be useful in many applications such as predictive learning of students, lie detectors, etc.

LIMITATIONS:

- The lack of common performances metric against which to evaluate new algorithms.
- Less accurate than deep learning-based techniques.
- Convolutional neural networks require a large amount of labeled training data to learn and generalize patterns accurately.
- Facial emotion datasets may not be representative of the entire population due to factors such as demographics, cultural differences, or imbalanced label distributions.
- Facial expressions are subjective and can vary across individuals, cultures, and contexts. A single emotion can be expressed differently by different people, making it challenging for the model to capture the full range of facial expression variations accurately.
- Facial emotion recognition solely relies on facial features, neglecting other contextual cues that influence emotion expression, such as body language, gestures, or situational context.
- Convolutional neural networks may struggle to detect and classify subtle or nuanced emotions, as they rely on prominent facial features.
- Convolutional neural networks trained on facial emotion datasets may not generalize well to unseen or novel data, especially if the data distribution differs significantly from the training set. This limitation can result in reduced performance when applied to real-world scenarios or unseen

individuals.

- Convolutional neural networks are often considered black box models, making it challenging to interpret and explain the decision-making process. Understanding the specific facial features or cues the model relies on for emotion recognition can be difficult, limiting the model's transparency and trustworthiness.
- Training and deploying convolutional neural networks for facial emotion recognition can be computationally demanding, requiring substantial computational resources and memory. This limitation can affect real-time applications or resource-constrained environments.

**[5] :TITLE:" EMOTION RECOGNITION FRO FACIAL EXPREEION
USING DEEP CNN"**

AUTHORS: D Y Liliana¹

JOURNAL NAME AND YEAR:2019

DESCRIPTION:

Automatic facial expression recognition is an actively emerging research in Emotion Recognition. This paper extends the deep Convolutional Neural Network (CNN) approach to facial expression recognition task. This task is done by detecting the occurrence of facial Action Units (AUs) as a subpart of Facial Action Coding System (FACS) which represents human emotion. In the CNN fully-connected layers we employ a regularization method called "dropout" that proved to be very effective to reduce overfitting. This research uses the extended Cohn Kanade (CK+) dataset which is collected for facial expression recognition experiment. The system performance gain average accuracy rate of 92.81%. The system has been successfully classified eight basic emotion classes. Thus, the proposed method is proven to be effective for emotion recognition.

LIMITATIONS:

- Processing every frame of video would be a huge job, thus typically only a small portion is subjected to a recognition system.
- It is still quite slow.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In the existing system, classification is done through simple image processing to classify images only.

Existing work includes the application of feature extraction of facial expressions with the combination of neural networks for the recognition of different facial emotions (happy, sad, angry, fear, surprised, neutral, etc.).

Humans are capable of producing thousands of facial actions during communication that varies in complexity, intensity, and meaning. The existing system is capable of analyzing the limitations of the existing system of Emotion recognition using brain activity.

3.1.1 DISADVANTAGE

- These systems only allow you to live a static user-experience as the system will give recommendation based on the history without regard to other parameters that might impact the prediction such as feeling or emotion.
- These recommendation systems will sometimes fail to give the correct output because their suggestions are based on outdated input.
- So, the user cannot be satisfied with the output as it doesn't satisfy his emotion.

3.2 PROPOSED SYSTEM

This paper proposes a system that has two main processes such as face detection and facial expression recognition (FER). This research focuses on an experimental study on identifying facial emotions. The flow for an emotion detection system includes the image acquisition, preprocessing of an image, face detection, feature extraction, and classification. To identify such emotions, the emotion detection system uses CNN Classifier for image classification, and Haar cascade algorithm an Object Detection Algorithm to identify faces in an image or a real-time video. This system works by taking live images from the webcam. The objective of this research is to produce an automatic facial emotion detection system to identify different emotions based on these experiments the system could identify several people that are sad, surprised, and happy, in fear, are angry, etc.

3.2.1 ADVANTAGES

- It will try to enhance the users' mood.
- Using facial emotion recognition can aid in understanding which emotions a user is experiencing in real-time.
- Testing of the system is done on the FER2013 dataset. Facial expressions are captured using an inbuilt camera.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 HARDWARE CONFIGURATION

- Processor: Pentium 4 (or equivalent)
- 4 GB RAM
- Hard disk space: 20 GB
- A projecting device (for the instructor only)
- A connection to the internet
- Keyboard and mouse or other pointing device

4.2 SOFTWARE REQUIREMENTS

- Operating System: Windows 10 version 1507
- Node.js
- Supported Internet browser: Chrome - Latest version, or the penultimate version

4.3 TECHNOLOGIES USED

4.3.1 Java Technology

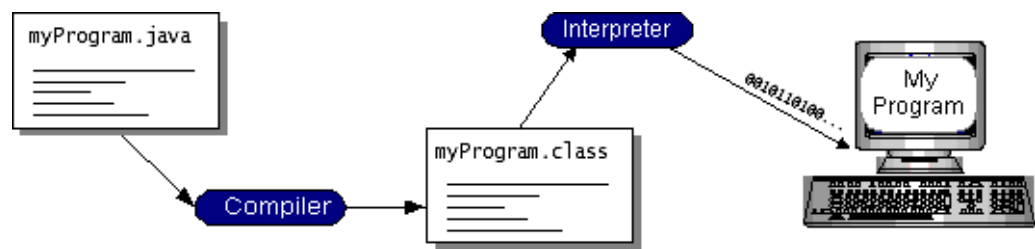
Java technology is both a programming language and a platform.

4.3.1.1 Java Programming Language

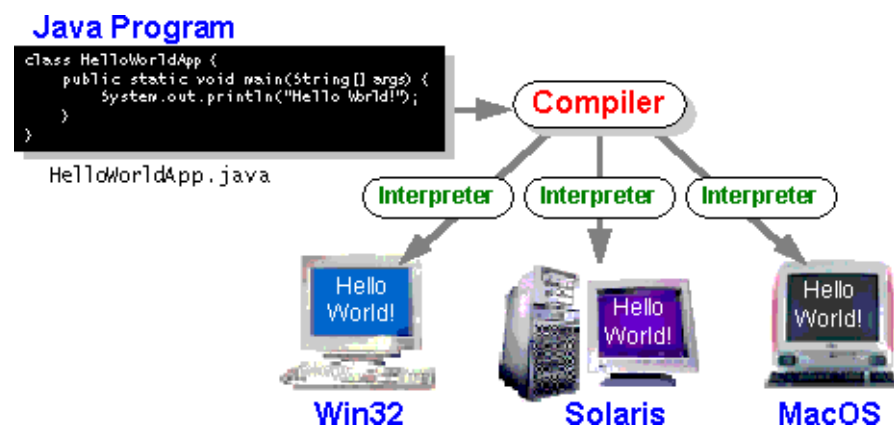
The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed.



You can think of Java byte codes as the machine code instructions for the *Java* Virtual Machine (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



4.3.1.2 Java Platform

A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware.

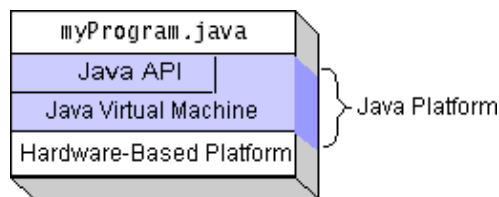
The Java platform has two components:

- The Java Virtual Machine (Java VM)
- The Java Application Programming Interface (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

What Can Java Technology Do?

The most common types of programs written in the Java programming language are applets and applications. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

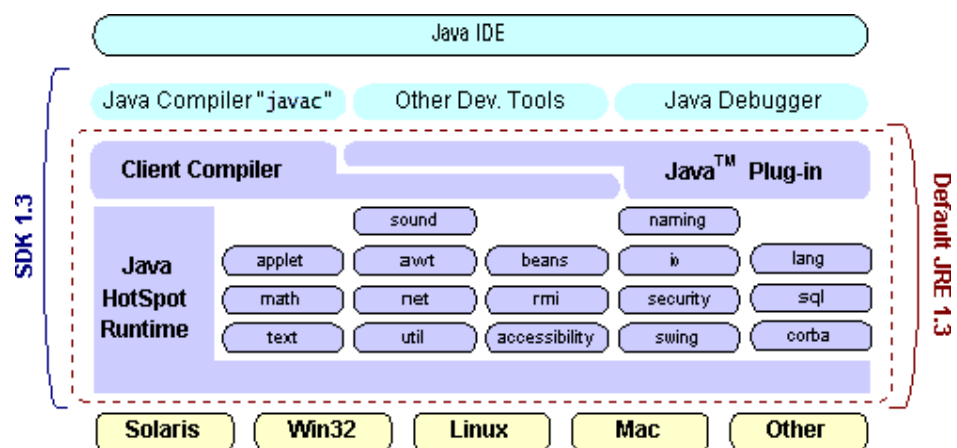
An application is a standalone program that runs directly on the Java platform. A special kind of application known as a server serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a servlet. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- **The essentials:** Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets:** The set of conventions used by applets.
- **Networking:** URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization:** Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- **Security:** Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components:** Known as JavaBeans, can plug into existing component architectures.
- **Object serialization:** Allows lightweight persistence and communication via Remote Method Invocation (RMI).

- **Java Database Connectivity (JDBC™):** Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

- **Get started quickly:** Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code:** Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code:** The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory.

its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.

- **Develop programs more quickly:** Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
- **Avoid platform dependencies with 100% Pure Java:** You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java™ Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
- **Write once, run anywhere:** Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily:** You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded “on the fly,” without recompiling the entire program.

4.3.1.3 ODBC :

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a de facto standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on

it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to

those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

4.3.1.4 JDBC

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight

as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

- **SQL Level API**

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

- **SQL Conformance**

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

- **JDBC must be implemental on top of common database interfaces**

The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

- **Provide a Java interface that is consistent with the rest of the Java system**

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

- **Keep it simple**

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate

functionality only serves to confuse the users of the API.

- **Use strong, static typing wherever possible**

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

- **Keep the common cases simple**

Because more often than not, the usual SQL calls used by the programmer are simple SELECT's, INSERT's, DELETE's and UPDATE's, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

Finally we decided to proceed the implementation using Java [Networking](#).

And for dynamically updating the cache table we go for MS [Access](#) database.

Java ha two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Simple	Architecture-neutral
Object-oriented	Portable
Distributed	High-performance
Interpreted	multithreaded
Robust	Dynamic
Secure	

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.

You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

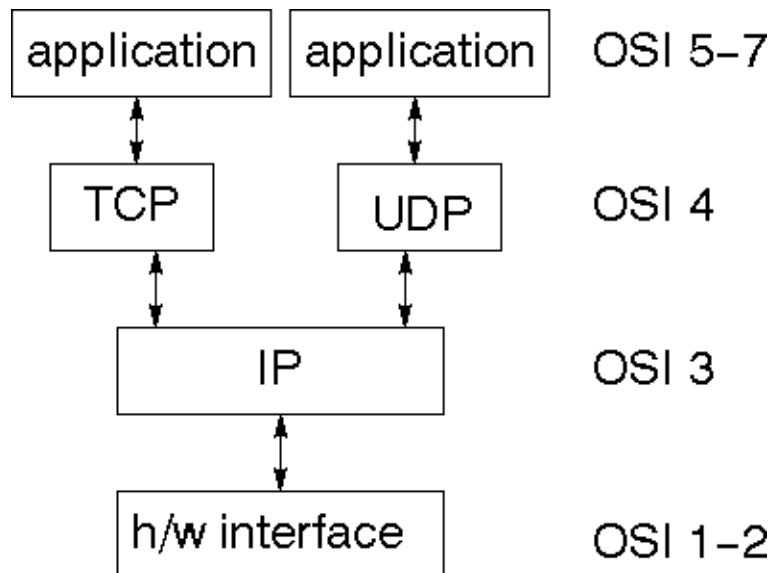
Java byte codes help make “write once, run anywhere” possible. You can

compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

4.3.1.5 Networking

TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

IP datagram's

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

UDP

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

TCP

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

Internet addresses

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

Network address

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

Subnet address

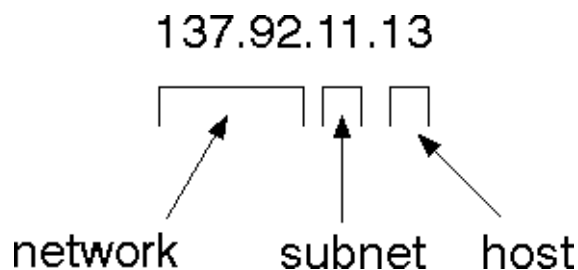
Internally, the UNIX network is divided into sub networks. Building 11 is

currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

Total address



The 32 bit address is usually written as 4 integers separated by dots.

Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call `socket`. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

```
#include <sys/types.h>
```

```
#include <sys/socket.h>
```

```
int socket(int family, int type, int protocol);
```

Here "family" will be `AF_INET` for IP communications, protocol will be

zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

4.3.1.6 JFree Chart

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

- A consistent and well-documented API, supporting a wide range of chart types;

- A flexible design that is easy to extend, and targets both server-side and client-side applications;

- Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

- JFreeChart is "open source" or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

- Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

- Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFreeChart;

Testing, documenting, testing some more, documenting some more.

2. Time Series Chart Interactivity

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

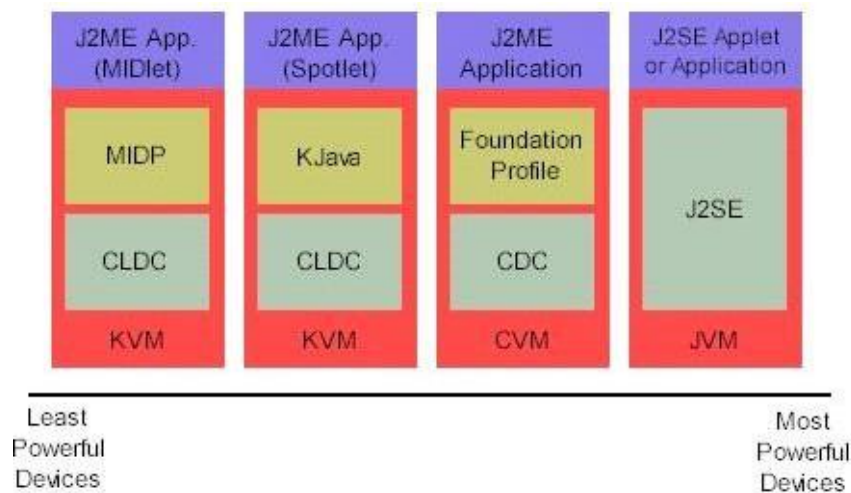
4. Property Editors

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

4.3.1.7 J2ME (Java 2 Micro edition):-

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the JavaOne Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

1. General J2ME architecture



J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the The profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the The following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine. While the J2SE virtual machine is generally referred to as a JVM, the J2ME virtual machines, KVM and CVM, are subsets of JVM. Both KVM and CVM can be thought of as a kind of Java virtual machine -- it's just that they are shrunken versions of the J2SE JVM and are specific to J2ME.

2. Developing J2ME applications

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices. We'll take a look at the way

the compiler is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role preverification plays in this process.

3. Design considerations for small devices

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "gotchas" before developing the application can be a painful process. Here are some design strategies to consider:

- * Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.
- * Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.
- * Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis. Other ways of reducing overall and peak memory use on small devices are to release resources quickly, reuse objects, and avoid exceptions.

4. Configurations overview

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

- * **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the

configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

5. J2ME profiles

What is a J2ME profile?

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: KJava and MIDP. Both KJava and MIDP are associated with CLDC and smaller devices. Profiles are built on top of configurations. Because profiles are specific to the size of the device (amount of memory) on which an application runs, certain profiles are associated with certain configurations.

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

Profile 1: KJava

KJava is Sun's proprietary profile and contains the KJava API. The KJava profile is built on top of the CLDC configuration. The KJava virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. KJava contains a Sun-specific API that runs on the Palm OS. The KJava API has a great deal in common with the J2SE Abstract Windowing Toolkit (AWT). However, because it is not a standard J2ME package, its main package is `com.sun.kjava`. We'll learn more

about the KJava API later in this tutorial when we develop some sample applications.

Profile 2: MIDP

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like KJava, is built upon CLDC and provides a standard run-time environment that allows new applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor. It is a complete and supported foundation for mobile application

development. MIDP contains the following packages, the first three of which are core CLDC packages, plus three MIDP-specific packages.

- * java.lang

- * java.io

- * java.util

CHAPTER 5

SYSTEM DESIGN SPECIFICATION

5.1 ARCHITECTURE DIAGRAM

In this system we use image or inbuilt camera images as an input image to determine the results. Firstly the 32 feature maps are introduced with each of containing unique features, which will double the size of the features by method of maxpooling based on the (average, maximum, or minimum) features, which passes through the fully connected layer, in this the each neuron is applies on a linear transformation to the input vector to predict which emotion has been givened with high accuracy.

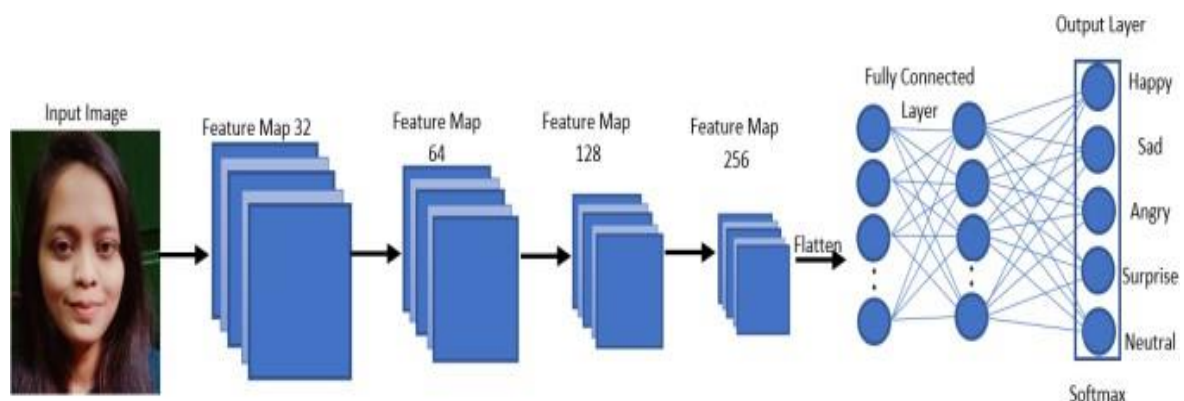
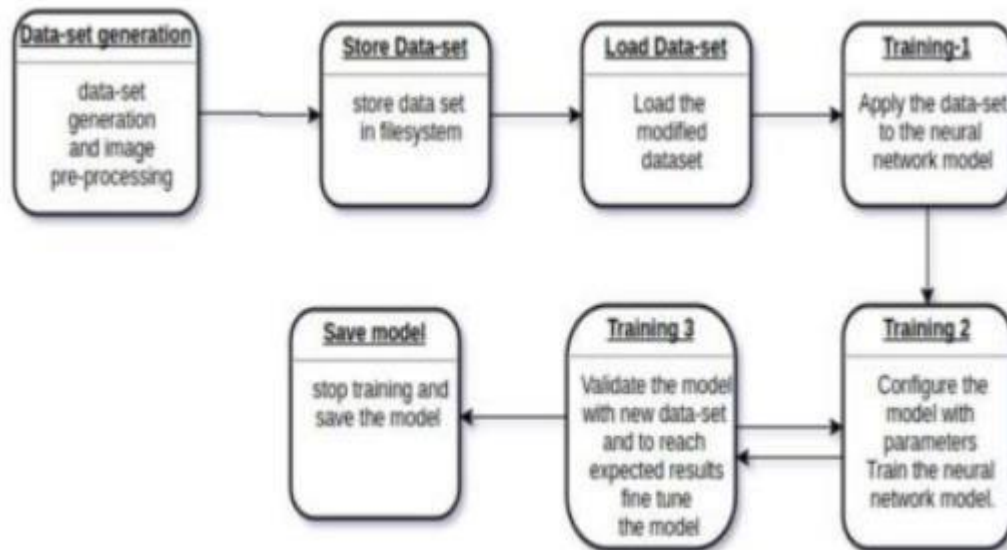


Fig 5.1-Architecture Diagram

5.2 DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement. They are often elements of a formal methodology such as Structured Systems Analysis and Design Method (SSADM). This data flow diagram shows the flow in performing secure

Data Flow diagram for Training



Data Flow diagram for Testing

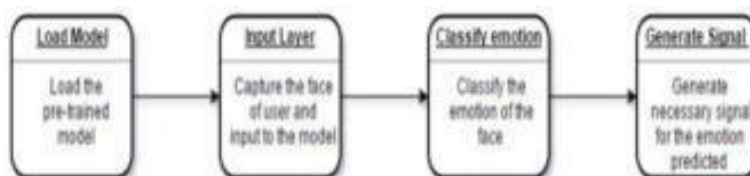


Fig 5.2 -Data flow diagram

5.3 USE CASE DIAGRAM

A use case is a set of scenarios that describe an interaction between a user and a system. A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use case and actors. An actor is representing a user or another system that will interact with the system modeled. A use case is an external view of

the system that represents some action the user might perform in order to complete a task.

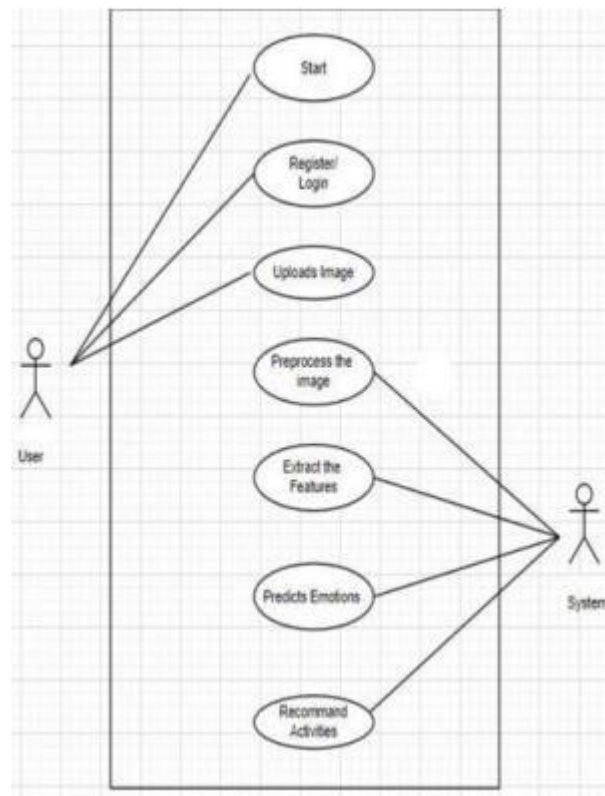
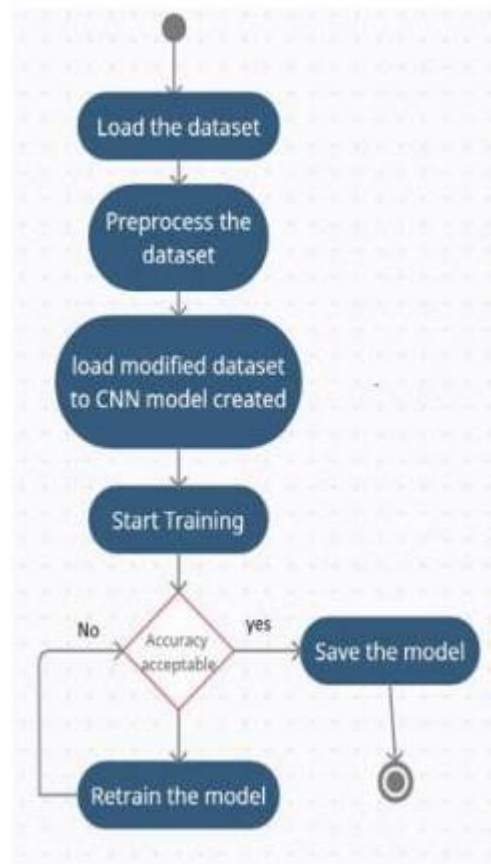


Fig 5.3 -Use case diagram

5.4 ACTIVITY DIAGRAM

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity described as an operation of the system.

Activity diagram for Training :



Activity Diagram for Testing:



Fig 5.4 -Activity diagram

5.5 SEQUENCE DIAGRAM

A sequence diagram is a diagram that illustrates the sequence of messages between objects in an interaction. A sequence diagram consists of a group of objects that are represented by lifelines, and the messages that they exchange over time during the interaction. Here this diagram shows the interaction between farmer, processor, distributor, retailer and consumer.

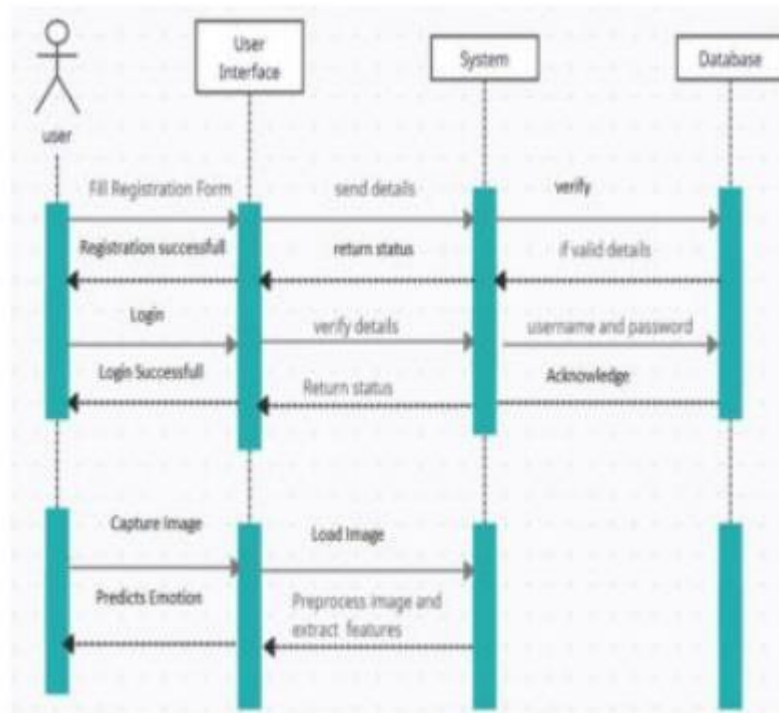


Fig 5.6-Sequence diagram

CHAPTER 6

SYSTEM DESIGN IMPLEMENTATION

6.1 IMPLEMENTATION

The implementation is done by giving the emotions like happy, sad, neutral, angry, surprise and our model have built is well and is able to make correct predictions based on given expressions with detailed high accuracy analysed by using the convolutional neural network (CNN) model and HAAR Cascade algorithm. It has performed several models to obtain better accuracy than the transfer learning model.

6.2 MODULE DESCRIPTION :

The modules in this project are

DATABASE DESCRIPTION

- Pre-Processing,
- Training Module,

EMOTION DETECTION

- Face Detection,
- Feature Extraction,
- Emotion Detection,

6.2.1 Database Description

We built the Convolutional Neural Network model using the Kaggle dataset. The database is FER2013 which is split into two parts training and testing dataset. The training dataset consists of 24176 and the testing dataset contains 6043 images. There are 48x48 pixel grayscale images of faces in the dataset. Each image in FER-2013 is labeled as one of five emotions: happy, sad, angry, surprise, and neutral. The faces

are automatically registered so that they are more or less centered in each image and take up about the same amount of space. The images in FER-2013 contain both posed and unposed headshots, which are in grayscale and 48x48 pixels.

The FER-2013 dataset was created by gathering the results of a Google image search of every emotion and synonyms of the emotions. FER systems being trained on an imbalanced dataset may perform well on dominant emotions such as happy, sad, angry, neutral, and surprised but they perform poorly on the under-represented ones like disgust and fear. Usually, the weighted-SoftMax loss approach is used to handle this problem by weighting the loss term for each emotion class supported by its relative proportion within the training set. However, this weighted-loss approach is predicated on the SoftMax loss function, which is reported to easily force features of various classes to stay apart without listening to intra-class compactness. One effective strategy to deal with the matter of SoftMax loss is to use an auxiliary loss to coach the neural network. To treating missing and Outlier values we have used a loss function named categorical crossentropy. For each iteration, a selected loss function is employed to gauge the error value. So, to treating missing and Outlier values, we have used a loss function named categorical crossentropy.



Figure 1. Samples from FER2013 dataset.

6.2.1.1 Pre-processing Module:

Following the capture of photos, we will do image processing on the captured images. The grey scale photos will be created by converting the colour photographs to grey scale.

6.2.1.2 Training module:

This step will involve the preparation of a dataset, which will consist of a binary array of all the photographs that have been taken. The collected photographs will be saved in a .YML file, which will contain all of the face data that was obtained. The .YML file allows us to process the collected photos more quickly because of its compressed nature.

6.2.2 Emotion Detection Module

6.2.2.1 Face Detection

Face detection is one of the applications which is considered under computer vision technology. This is the process in which algorithms are developed and trained to properly locate faces or objects in object detection or related system in images. This detection can be real-time from a video frame or images. Face detection uses such classifiers, which are algorithms that detect what's either a face (1) or not a face (0) in an image. Classifiers are trained to detect faces using numbers of images to get more accuracy. OpenCV uses two sorts of classifiers, LBP (Local Binary Pattern) and Haar Cascades. A Haar classifier is used for face detection where the classifier is trained with pre-defined varying face data which enables it to detect different faces accurately. The main aim of face detection is to spot the face within the frame by reducing external noises and other factors. It is a machine learning-based approach where the cascade function is trained with a group of input files. It is supported the

Haar Wavelet technique to research pixels inside the image into squares by function [9]. This uses machine learning techniques to urge a high degree of accuracy from what's called "training data".

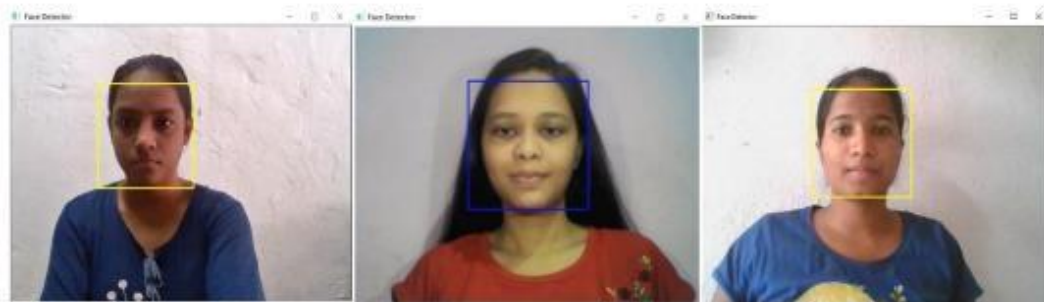


Figure 2. Face detection

6.2.2.2 Feature Extraction

While performing feature extraction, we treat the pre-trained network that is a sequential model as an arbitrary feature extractor. Allowing the input image to pass on it forward, stopping at the pre-specified layer, and taking the outputs of that layer as our features. Starting layers of a convolutional network extract high-level features from the taken image, so use only a few filters. As we make further deeper layers, we increase the number of the filters to twice or thrice the dimension of the filter of the previous layer. Filters of the deeper layers gain more features but are computationally very intensive.

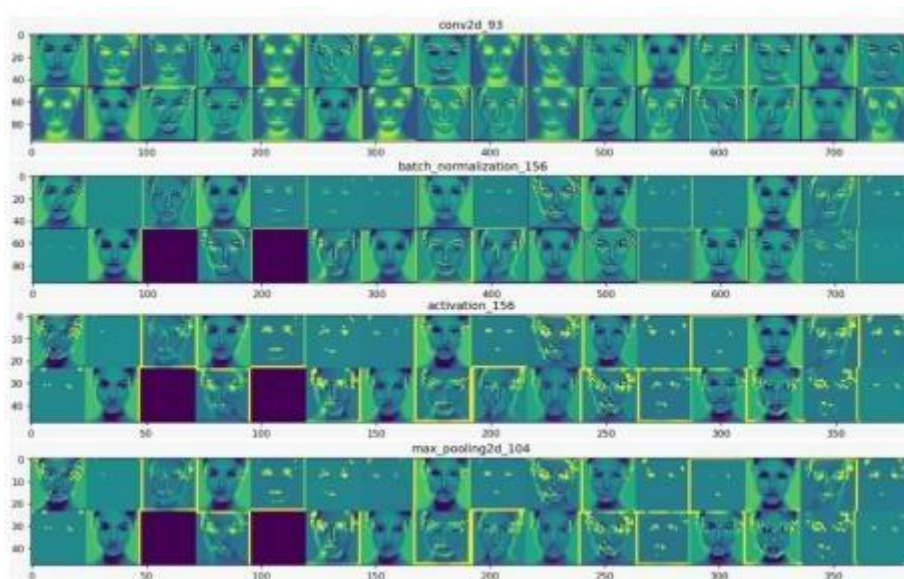


Figure 3. Visualization of The Feature Map

Doing this we utilized the robust, discriminative features learned by the Convolution neural network [10]. The outputs of the model are going to be feature maps, which are an intermediate representation for all layers after the very first layer. Load the input image for which we want to view the Feature map to know which features were prominent to classify the image. Feature maps are obtained by applying Filters or Feature detectors to the input image or the feature map output of the prior layers. Feature map visualization will provide insight into the interior representations for specific input for each of the Convolutional layers within the model.

6.2.2.3 Emotion Detection

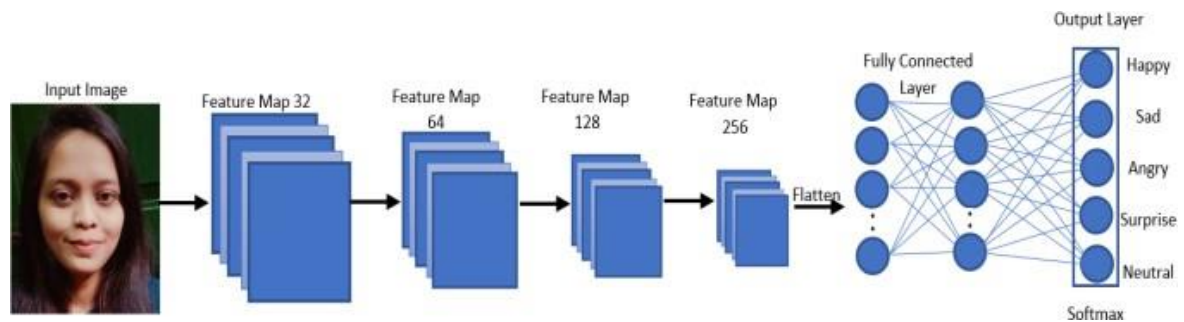


Figure 4. Convolution neural network Architecture

Convolution neural network architecture applies filters or feature detectors to the input image to get the feature maps or activation maps using the Relu activation function [11]. Feature detectors or filters help in identifying various features present in the image such as edges, vertical lines, horizontal lines, bends, etc. After that pooling is applied over the feature maps for invariance to translation. Pooling is predicted on the concept that once we change the input by a touch amount, the pooled outputs don't change. We can use any of the pooling from min, average, or max. But max-pooling provides better performance than min or average pooling. Flatten all the input and giving these flattened inputs to a deep neural network which are outputs to the class of the object.

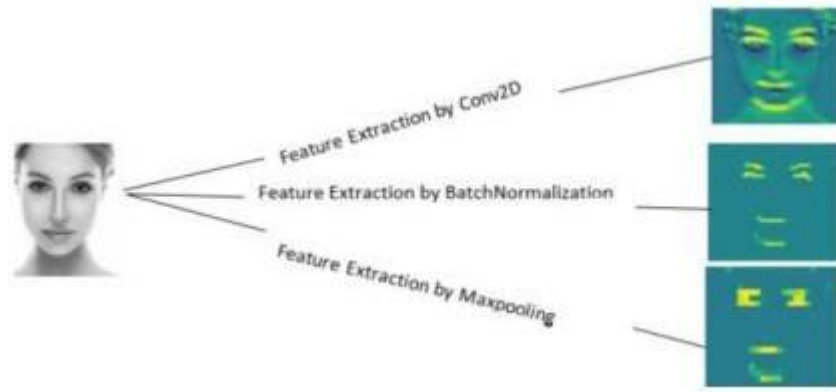


Figure 5. Feature Extraction by each layer in Convolutional Neural Network

The class of the image will be binary, or it will be a multi-class classification for identifying digits or separating various apparel items. Neural networks are as a black box, and learned features in a Neural Network are not interpretable. So basically, we give an input image then the CNN model returns the results [10]. Emotion detection is performed by loading the model which is trained by weights using CNN. When we take the real-time image by a user then that image was sent to the pre-trained CNN model, then predict the emotion and adds the label to the image.



Figure 6. Results of Emotion Detection.

CHAPTER 7

SYSTEM TESTING AND MAINTENANCE

7.1 TESTING OBJECTIVES

In the testing process we test the actual system in an organization and gather errors from the new system and take initiatives to correct the same. All the front-end and back-end connectivity are tested to be sure that the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently.

The main objective of testing is to uncover errors from the system. For the uncovering process we have to give proper input data to the system. So we should be more conscious to give input data. It is important to give correct inputs to efficient testing. Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects. Thus the system testing is a confirmation that all is correct and an opportunity to show the user that the system work.

7.2 TESTING AND METHODOLOGY

7.2.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.2.2 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes

must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

7.2.3 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.2.4 White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

7.2.5 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

7.2.6 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.2.7 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENT

8.1 CONCLUSION

We propose a face expression identification approach based on a CNN model that effectively extracts facial features in this research. The suggested method uses training sample image data to directly input the picture pixel value. The ability to accurately determine emotions was greatly enhanced by the removal of the background. Emotion expression is important in communication, hence improving the quality of interaction between humans. Furthermore, in the near future, the study of facial expression detection may provide improved feedback to society as well as the interaction between Human-Robot interfaces (HRI). Emotion detection mostly involves the geometric part of the face (e.g.; eyes, eyebrows, and mouth). The review takes into consideration of experiments which been conducted in a controlled environment, in real-time, and in wild images. The recent research, particularly the performance with profile views, will be applicable to a greater range of realworld commercial applications, such as patient monitoring in a hospital or surveillance security. Furthermore, the concept of facial emotion recognition could be expanded to include emotion recognition from speech or body motions in order to address emerging industrial applications.

8.2 FUTURE ENHANCEMENT

An accumulation is often needed to gather the partial results from these parallel executions in different servers. The runtime system captures new events and run corresponding actions to analyze the page and store more URLs into the URL set to generate new events.

APPENDIX 1

SOURCE CODE

SAMPLE CODING

```
"name": "facial-emotion-detector",
"version": "0.1.0",
"private": true,
"dependencies": {
  "@testing-library/jest-dom": "^4.2.4",
  "@testing-library/react": "^9.5.0",
  "@testing-library/user-event": "^7.2.1",
  "gh-pages": "^3.1.0",
  "react": "^16.13.1",
  "react-dom": "^16.13.1",
  "react-scripts": "^3.4.4"
},
"homepage": "http://localhost:3000",
"scripts": {
  "start": "react-scripts start",
  "build": "react-scripts build",
  "test": "react-scripts test",
  "eject": "react-scriptseject",
  "deploy": "npm run build && gh-pages -d build"
},
"eslintConfig": {
  "extends": "react-app"
},
"browserslist": {
  "production": [
    ">0.2%",
```

```

"notdead",
"notop_miniall"
],
"development":[
"last1chromeversion",
"last1firefoxversion",
"last1safari1version"
]
}
}
{
"name":"facial-emotion-detector",
"version":"0.1.0",
"lockfileVersion":2,
"requires":true,
"packages":{
"":{
"name":"facial-emotion-detector",
"version":"0.1.0",
"dependencies":{
"@testing-library/jest-dom":"^4.2.4",
"@testing-library/react":"^9.5.0",
"@testing-library/user-event":"^7.2.1",
"gh-pages":"^3.1.0",
"react":"^16.13.1",
"react-dom":"^16.13.1",
"react-scripts":"^3.4.4"
}
},
"node_modules/@babel/code-frame":{

```

```

"version":"7.10.4",
"resolved":"https://registry.npmjs.org/@babel/code-frame/-/code-frame-7.10.4.tgz",
"integrity":"sha512-
vG6SvB6oYEhvgisZNFRmRCUkLz11c7rp+tbNTynGqc6mS1d5ATd/sGyV6W0K
ZZnXRKMTzZ
DRgQT3Ou9jhpAfUg==",
"dependencies":{
"@babel/highlight":"^7.10.4"
},
"node_modules/@babel/compat-data":{
"version":"7.11.0",
"resolved":"https://registry.npmjs.org/@babel/compat-data/-/compat-data-
7.11.0.tgz",
"integrity":"sha512-
TPSvJfv73ng0pfNEOh17bYMPQbI95+nGWc71Ss4vZdRBHTDqmM9Z8ZV4rYz8
Ks7sfzc95n3
0k6ODIq5UGnXcYQ==",
"dependencies":{
"browserslist":"^4.12.0",
"invariant":"^2.2.4",
"semver":"^5.5.0"
},
"node_modules/@babel/compat-data/node_modules/semver":{
"version":"5.7.1",
"resolved":"https://registry.npmjs.org/semver/-/semver-5.7.1.tgz",
"integrity":"sha512-
sauaDf/PZdVgrLTNYHRtpXa1iRiKcaebiKQ1BJdpQlWH2lCvexQdX55snPFyK7Q
zpudqbCI0qX

```

```

FfOasHdyNDGQ==",
"bin":{
"semver":"bin/semver"
}
},
"node_modules/@babel/core":{
"version":"7.9.0",
"resolved":"https://registry.npmjs.org/@babel/core/-/core-7.9.0.tgz",
"integrity":"sha512-
kWc7L0fw1xwvI0zi8OKVBuxRVefwGOrKSQMvrQ3dW+bIIavBY3/NpXmpjMy
7bQnLgwgzWQZ
8TIM57YHpHNHz4w==",
"dependencies":{
"@babel/code-frame":"^7.8.3",
"@babel/generator":"^7.9.0",
"@babel/helper-module-transforms":"^7.9.0",
"@babel/helpers":"^7.9.0",
"@babel/parser":"^7.9.0",
"@babel/template":"^7.8.6",
"@babel/traverse":"^7.9.0",
"@babel/types":"^7.9.0",
"convert-source-map":"^1.7.0",
"debug":"^4.1.0",
"gensync":"^1.0.0-beta.1",
"json5":"^2.1.2",
"lodash":"^4.17.13",
"resolve":"^1.3.2",
"semver":"^5.4.1",
"source-map":"^0.5.0"
},

```

```

"engines":{
"node":">>=6.9.0"
},
"funding":{
"type":"opencollective",
"url":"https://opencollective.com/babel"
}
},
"node_modules/@babel/core/node_modules/semver":{
"version":"5.7.1",
"resolved":"https://registry.npmjs.org/semver/-/semver-5.7.1.tgz",
"integrity":"sha512-sauaDf/PZdVgrLTNYHRtpXa1iRiKcaebiKQ1BJdpQlWH2lCvexQdX55snPFyK7Q
zpuDqbCI0qX
FfOasHdyNDGQ==",
"bin":{
"semver":"bin/semver"
}
},
"node_modules/@babel/generator":{
"version":"7.11.0",
"resolved":"https://registry.npmjs.org/@babel/generator/-/generator-7.11.0.tgz",
"integrity":"sha512-fEm3Uzw7Mc9Xi//qU20cBKatTfs2aOtKqmvv/Vm7RkJEGFQ4xc9myCfbXxqK//Z
S8MR/ciOH
w6meGASJuKmDfQ==",
"dependencies":{
"@babel/types":"^7.11.0",
"jsesc":"^2.5.1",
"source-map":"^0.5.0"
}
}

```

```

}
},
"node_modules/@babel/helper-annotate-as-pure":{
"version":"7.10.4",
"resolved":"https://registry.npmjs.org/@babel/helper-annotate-as-pure/-/helper-
annotate
-as-pure-7.10.4.tgz",
"integrity":"sha512-
XQlqKQP4vXFB7BN8fEEerrmYvHp3fK/rBkRFz9jaJbzK0B1DSfej9Kc7ZzE8Z/O
nId1jpJdNAZ3
BFQjWG68rcA==",
"dependencies":{
"@babel/types":"^7.10.4"
}
},
"node_modules/@babel/helper-builder-binary-assignment-operator-visitor":{
"version":"7.10.4",
"resolved":"https://registry.npmjs.org/@babel/helper-builder-binary-assignment-
operator
-visitor/-/helper-builder-binary-assignment-operator-visitor-7.10.4.tgz",
"integrity":"sha512-
L0zGIFrGWZK4PbT8AszSfLTM5sDU1+A/En9VrdT8/LmEiJt4zXt+Jve9DCAnQ
cbqDhCI+29y/
L93mrDzddCc==",
"dependencies":{
"@babel/helper-explode-assignable-expression":"^7.10.4",
"@babel/types":"^7.10.4"
}
},
"node_modules/@babel/helper-builder-react-jsx":{

```



```

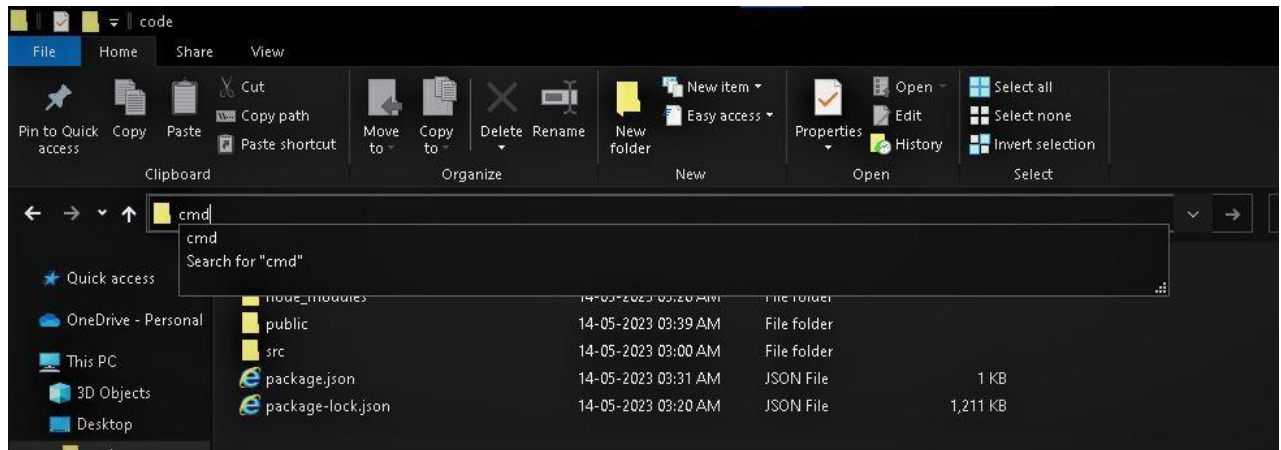
"version":"7.10.4",
"resolved":"https://registry.npmjs.org/@babel/helper-builder-react-jsx/-/helper-builderreact-jsx-7.10.4.tgz",
"integrity":"sha512-5nPcIZ7+KKDxT1427oBivl9V9YTal7qk0diccnh7RrcgrT/pGFOjgGw1dgryyx1GvHEpXVfoDF6A
k3rTiWh8Rg==",
"dependencies":{
"@babel/helper-annotate-as-pure":"^7.10.4",
"@babel/types":"^7.10.4"
},
"node_modules/@babel/helper-builder-react-jsx-experimental":{
"version":"7.10.5",
"resolved":"https://registry.npmjs.org/@babel/helper-builder-react-jsx-experimental/-
/helper-builder-react-jsx-experimental-7.10.5.tgz",
"integrity":"sha512-Buewnx6M4ttG+NLkKyt7baQn7ScC/Td+e99G914fRU8fGIUivDDgVIQeDHF5e4CRSJQt58Wp
NHhsAZgtzVhsg==",
"dependencies":{
"@babel/helper-annotate-as-pure":"^7.10.4",
"@babel/helper-module-imports":"^7.10.4",
"@babel/types":"^7.10.5"
},
},

```

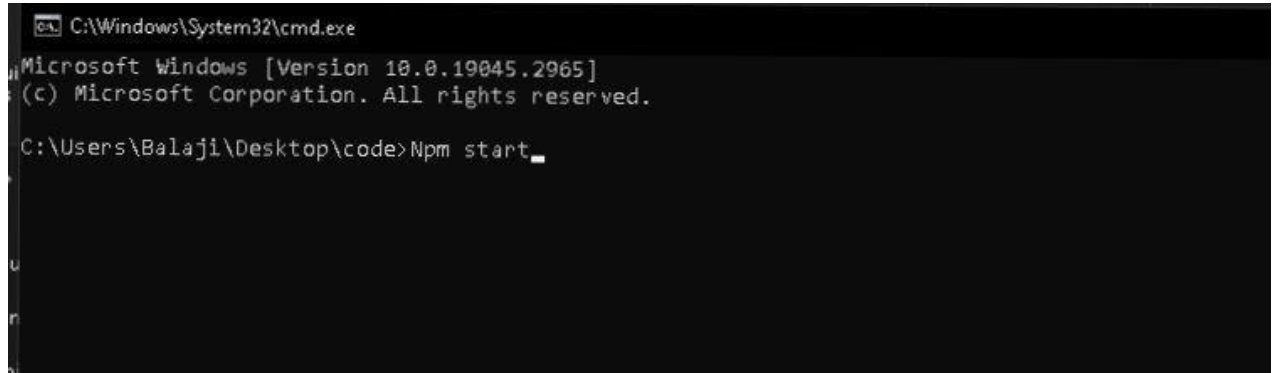
APPENDIX 2

OUTPUT SCREENSHOTS

Initial phase

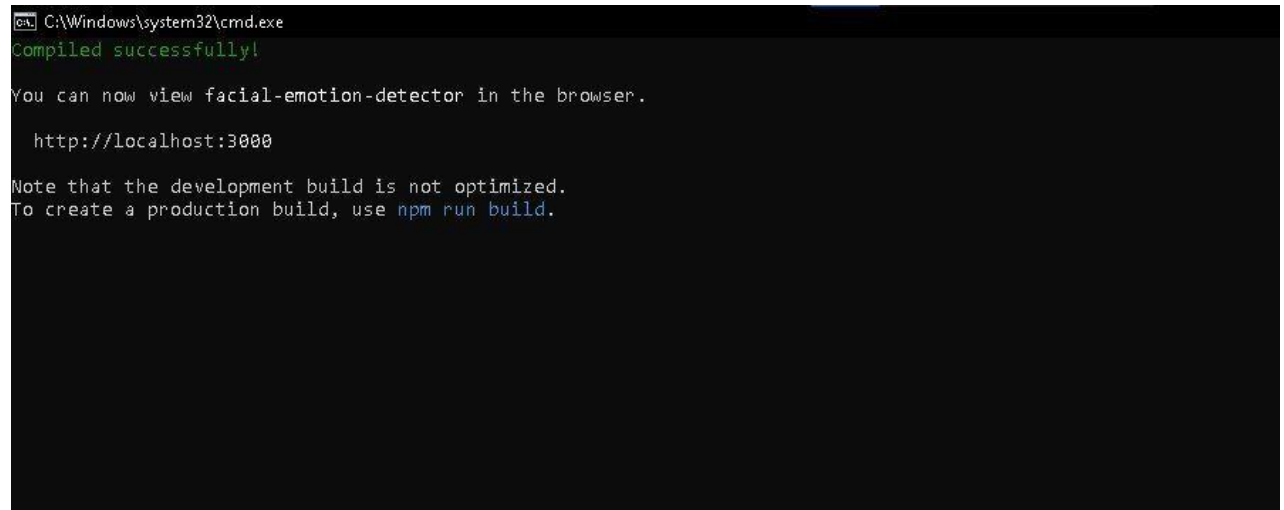


Start command



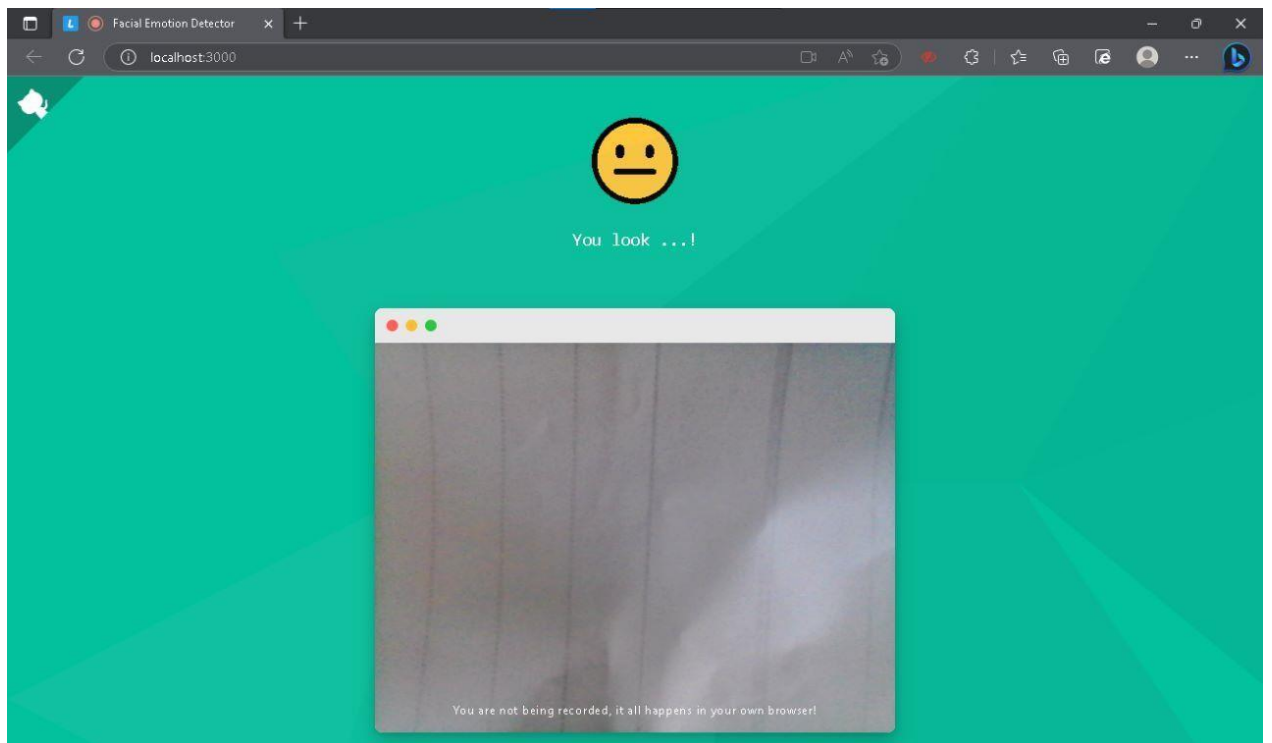
```
C:\Windows\system32\cmd.exe
? Something is already running on port 3000.
Would you like to run the app on another port instead? (Y/n) y
```

Complete successfully

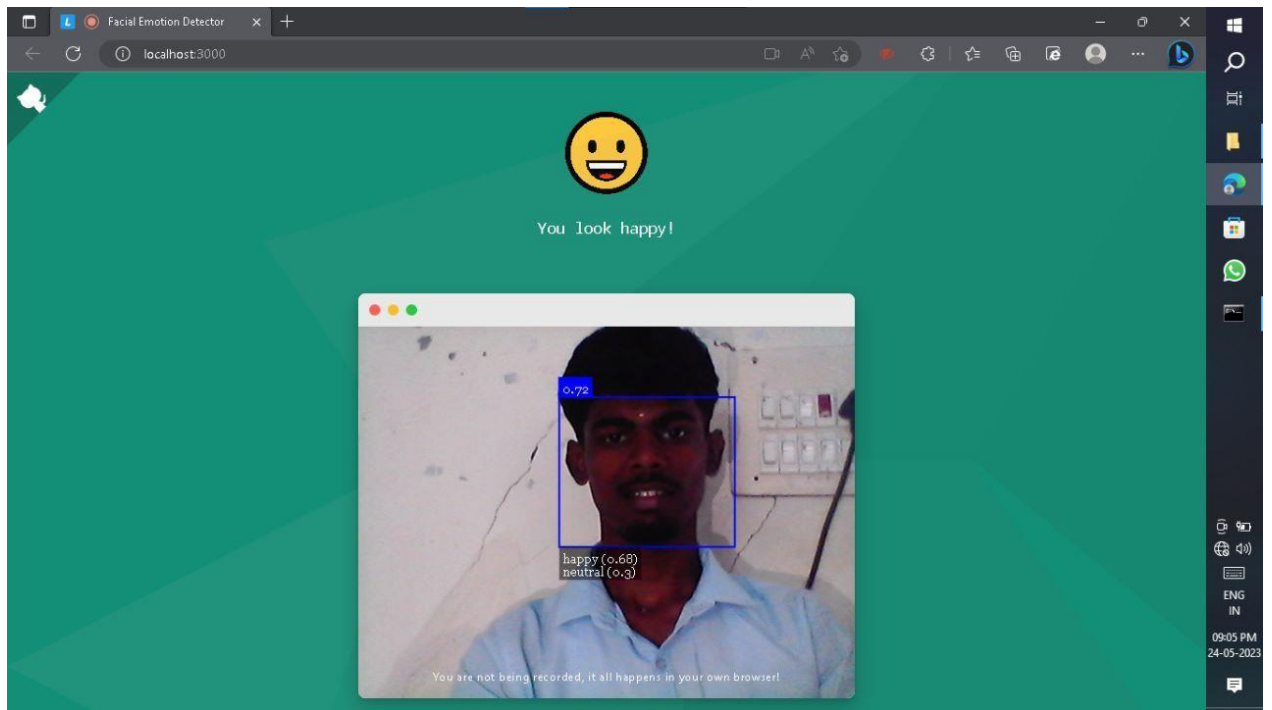
A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\system32\cmd.exe'. The terminal text is as follows:

```
Compiled successfully!  
  
You can now view facial-emotion-detector in the browser.  
  
  http://localhost:3000  
  
Note that the development build is not optimized.  
To create a production build, use npm run build.
```

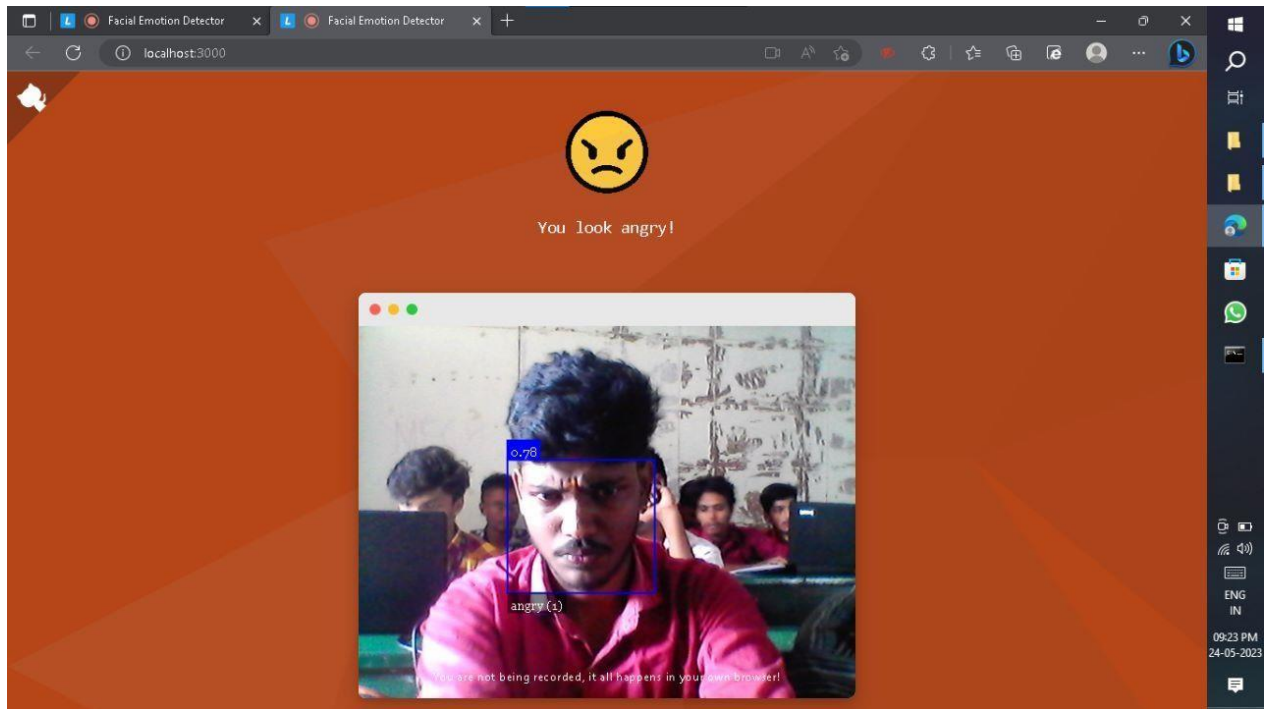
OUTPUT PAGE



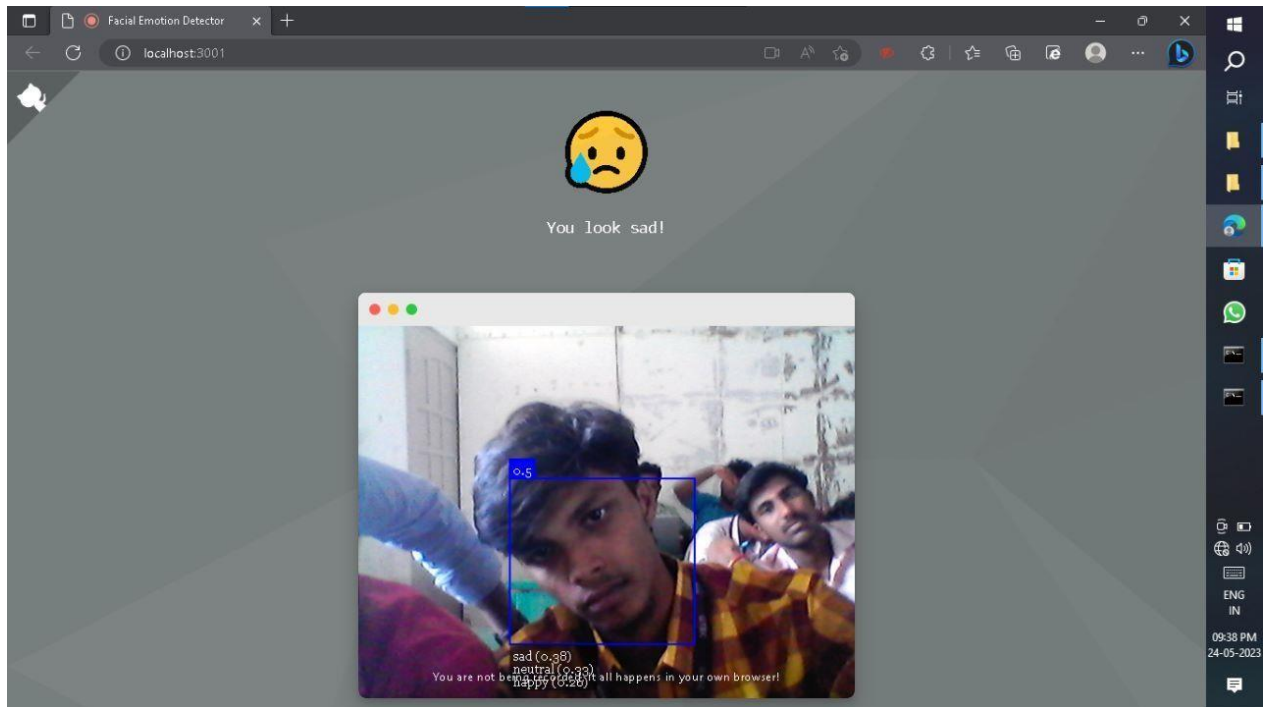
HAPPAY FACE



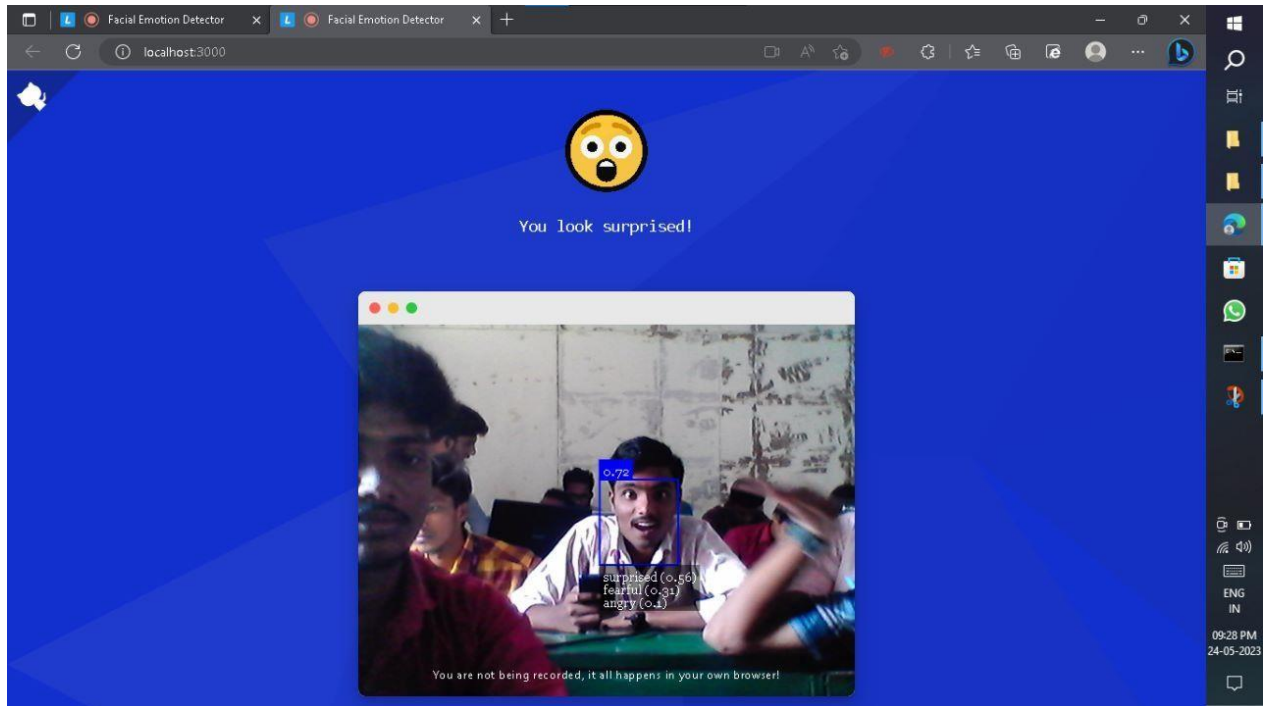
ANGRY FACE



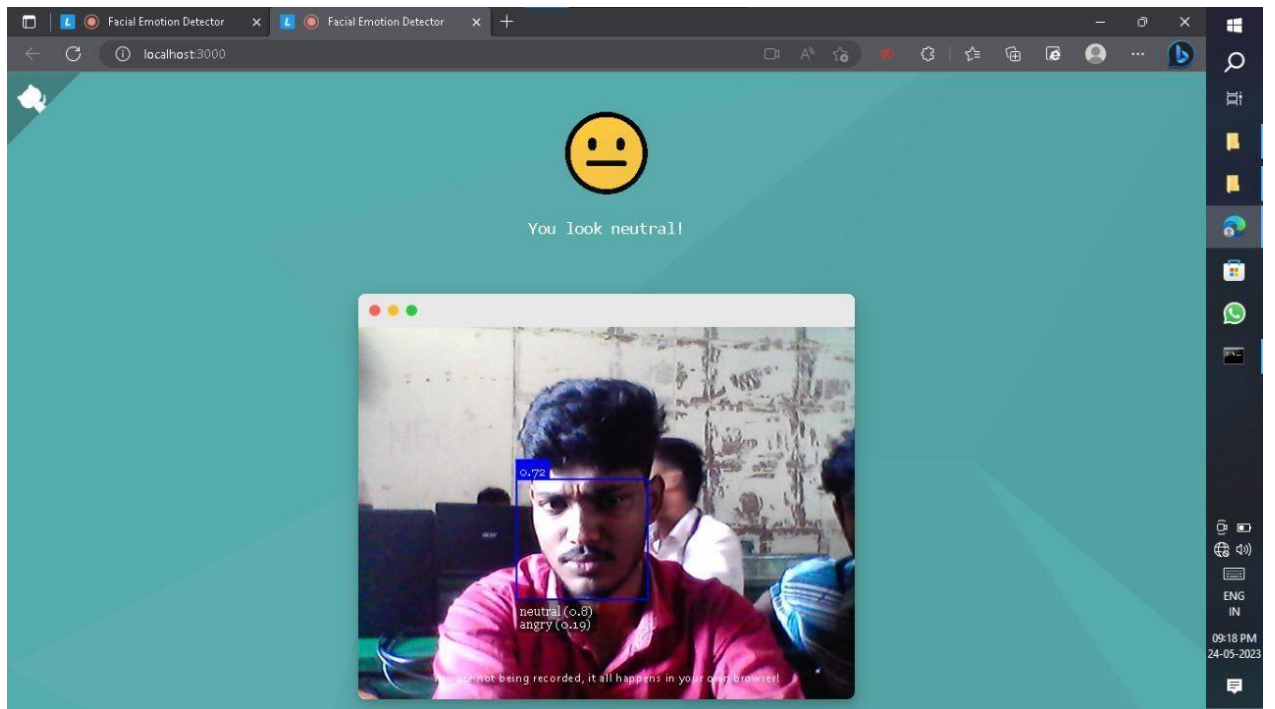
SAD FACE



SURPRISED FACE



NEUTRAL FACE



REFERENCES

- K. F. Azizan Illiana, "Facial Emotion Recognition: A Brief Review," in International Conference on Sustainable Engineering, Technology and Management 2018 (ICSETM-2018), 2020.
- [2] R. Shyam, "Convolutional Neural Network and its Architectures." Journal of Computer Technology & Applications, vol. 12, no. 2, pp. 6-14, 2021.
- [3] R. Shyam, "Machine Learning and Its Dominant Paradigms," Journal of Advancements in Robotics, vol. 8, no. 2, pp. 1-10, 2021.
- [4] R. Shyam, "Automatic Face Recognition in Digital World," Advances in Computer Science and Information Technology (ACSIT), vol. 2, no. 1, pp. 64-70, 2015.
- [5] S. R. N. S. M. A. H. Akhand, "Facial Emotion Recognition Using Transfer Learning in the Deep CNN," MDPI, vol. 10, no. 9, 2021.
- [6] N. Mehendale, "Facial emotion recognition using convolutional neural networks (FERC)," SN Applied Sciences, vol. 2, no. 3, 2020.
- [7] N. R. S, "Emotion Recognition from Facial Expression using deep learning," International Journal of Engineering and Advanced Technology (IJEAT), vol. 8, no. 6S, 2019.
- [8] R. Shyam, "Enhanced Object Detection with Deep Convolutional Neural Networks," International Journal of All Research Education and Scientific Methods (IJARESM), vol. 9, no. 7, pp. 27-36, 2021.